INNOVATION & DEVELOPMENT

LABS

DEEP LEARNING

Architectures and Applications

Binal Patel

binal.patel@mu-sigma.com

Innovation Analyst/Data Analyst

Mu Sigma

ABSTRACT

This paper will provide a broad overview of deep learning: what it is, how it is structured, how models' learn, and how to use it in the real-world. It will especially focus on the application of deep learning to natural language processing (NLP) and sentiment analysis. Deep learning has been proven to outperform classical machine learning in almost every area, and this paper sets out to explain why this is so.

INTRODUCTION

Deep learning refers to a class of machine learning techniques that are analogous to how the human brain learns something. For example, a child doesn't learn how to speak their mother tongue immediately. Instead, the child first learns the basic noises of the language, the phonemes. They learn how to mumble and utter sounds. They then learn how to piece together these sounds into a short word, such as "no". Soon thereafter they learn to piece together the words into short sentences, and the sentences into complete thoughts, and so on.

The same principles underlie deep learning. Deep learning involves using many layers of information processing in hierarchical architectures. As data is passed from layer to layer, the network learns progressively more complex representations. Let's use the example of a deep network trained to recognize speech. Simplistically, the first layer would learn how to recognize basic phonemes, the second could learn to recognize words, the third could learn to recognize phrases, the next sentences, and so on. Once fully trained, the network would be able to distinguish between words. Google, in fact, uses deep networks in this fashion to train the voice recognition within their Android platform (Google).

This technique has proven itself to be very powerful. It extends the basic premise of machine learning, being able give a computer program the ability to learn without being explicitly programmed, to the next level. The computer can now not only learn, it can also combine concepts to learn better, which is much closer to how humans learn.

For example, a common machine learning task is to be able to recognize written text, commonly used within "captcha" systems for verification. However, that's all that they can do. A deep network would be able to recognize written text, parse it together as a sentence, and be able to check if the sentence is a valid English sentence or not.

Below is an example of a deep learning network that Google trained. It was able to differentiate between humans and felines by hierarchically learning more complex features from layer to layer. As seen below, it first learned how to recognize simple lines, than shapes, until it could recognize human faces and feline faces.

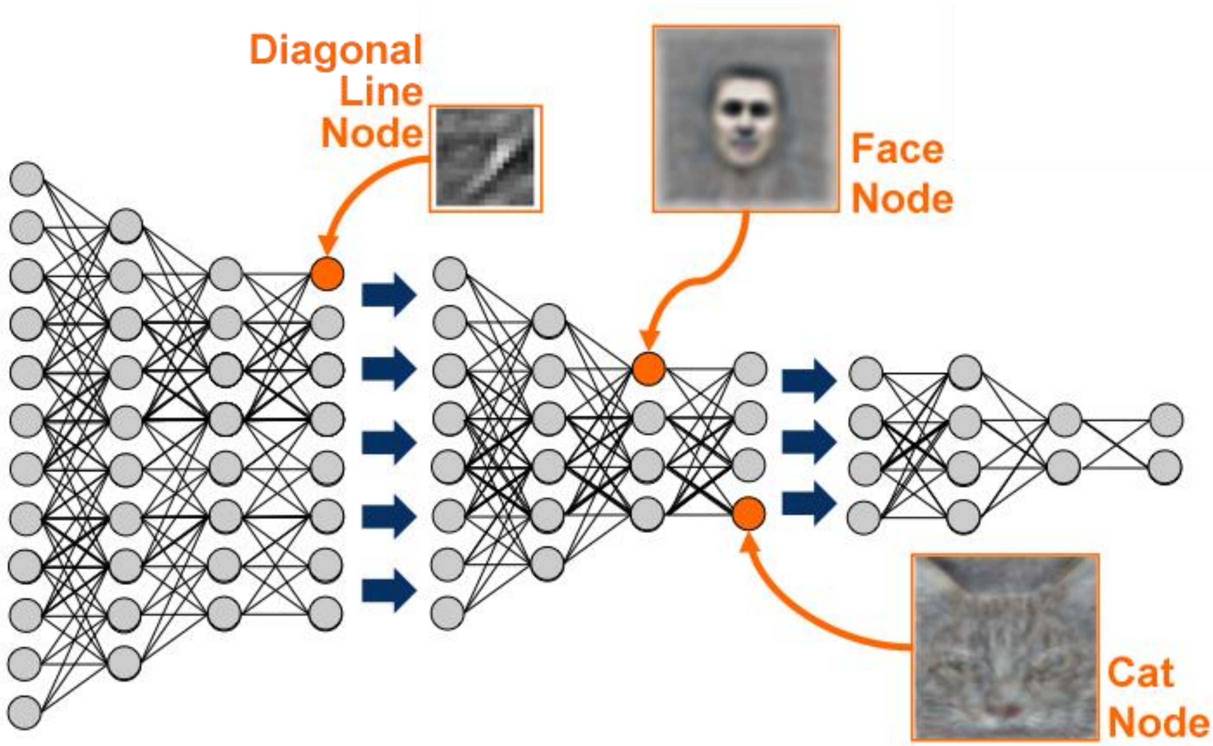

Figure 1 - Deep Learning Network for Image Recognition

NEURAL NETWORKS

Structure

A substantial number of deep learning models are variants of neural networks. Neural networks are models inspired by the central nervous system; by how neurons interact with each other to pass on electrical signals. Simplistically, electrical signals form a variety of neurons are sent to another neuron, and this neuron fires once the combination of signals reaches a certain threshold. This neuron fires a signal to another neuron, where the same process occurs once more.

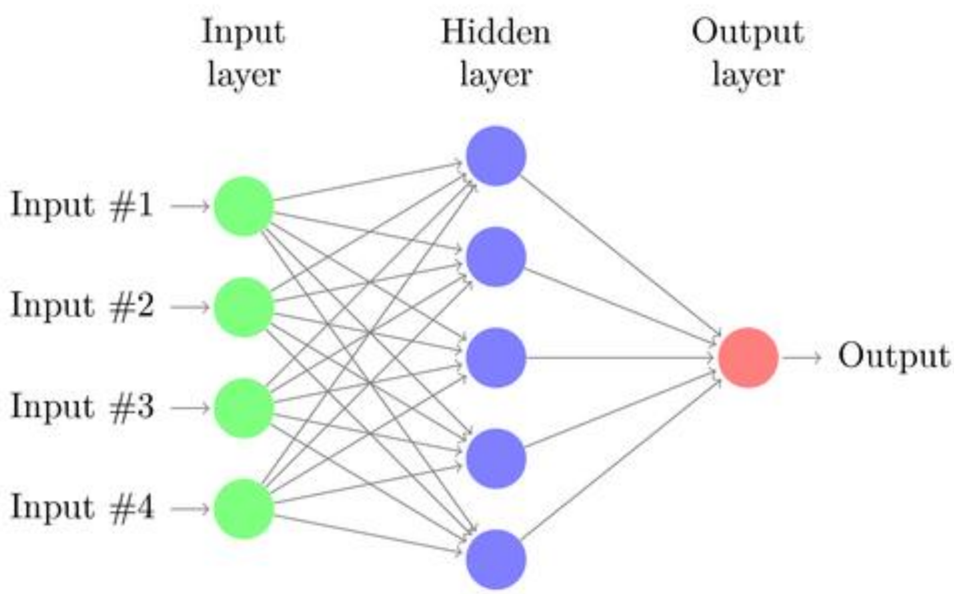

Figure 2 - Feed Forward Neural Network

A simple feed-forward neural network is shown above. Input is fed into a hidden layer, and the hidden layer is fed into an output layer. This is commonly referred to as "feeding forward" the data. Each node in the hidden layer and output layer represents a neuron, and is composed of two portions. One part is a summation function, which simple sums up the weights of all incoming signals multiplied by all incoming inputs. This segment is called the "transfer function".

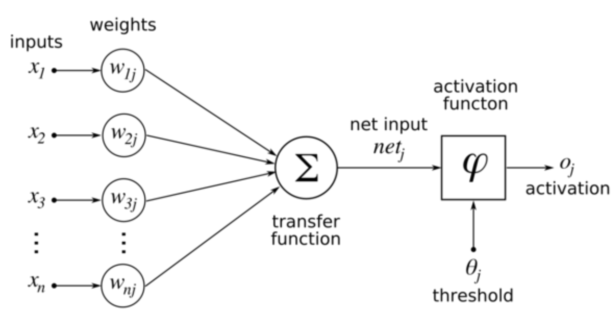

Figure 3 - Composition of a neuron

The transfer function sends its' output to an activation function, which models the actual neuron. Modern neural networks differ drastically from the biological model in that the activation function is usually a sigmoid function, such as the logistic function or hyperbolic tan function. Thus, no threshold is set, and the neuron always outputs a real-valued number been 0 and 1, or -1 and -1. This allows the network to learn how to solve problems that are not linearly separable, such as the exclusive-or problem. A non-linear activation function is also important to train the neural network, since every sigmoid function is easily differentiable.

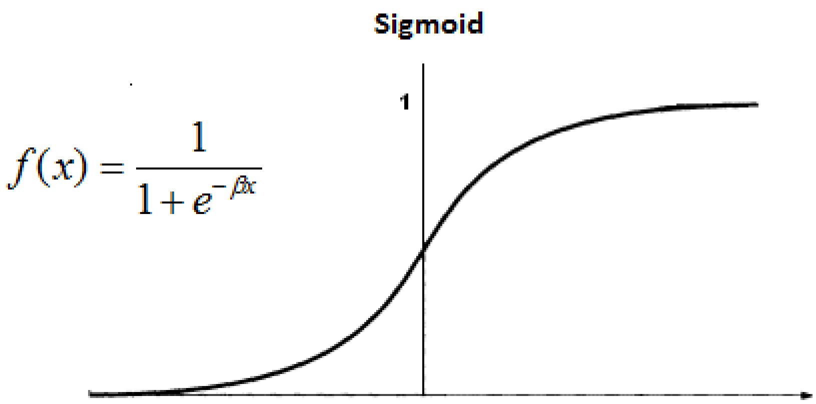

Figure 4 - The normal activation function.

Thus, data is fed forward in this fashion, from the input layer to the hidden layer, and from the hidden layer to the output layer, which results in a certain numerical output. Neural networks are initialized with a predetermined number of neurons in each layer, and with randomly generated weights for every connection between neurons. Therefore, the output in a simple forward pass is nonsensical and useless. To generate a useful neural network, it must be trained. (More specifically, the weights between neurons must be tuned to generate useful results.)

Training

Neural networks are trained using various algorithms that sequentially update the weights between the various neurons. The most common algorithm used is the backpropagation algorithm, which is short hand for "backpropagation of errors." The output of the neural network is compared against the targeted output, and the error signal from this is propagated backwards through the network, from the output layer, to the hidden layer, to the input layer. Weights between the layers are then updated accordingly.

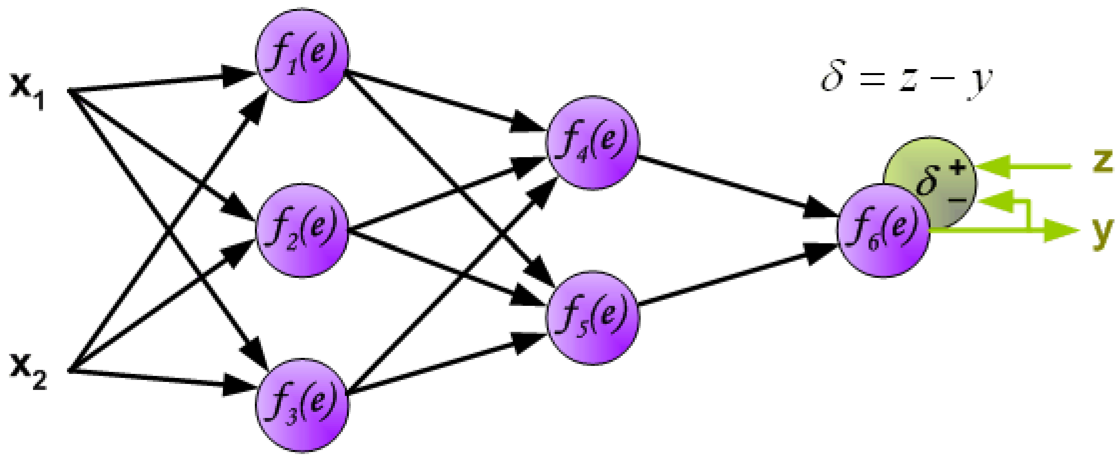

Figure 5 - Calculation of the error signal.

The error is simply the difference between the target output, and the actual output. This error is sent backwards through the network using existing weights. An error signal is calculated for each neuron by using the existing weights to send it backwards through the network.

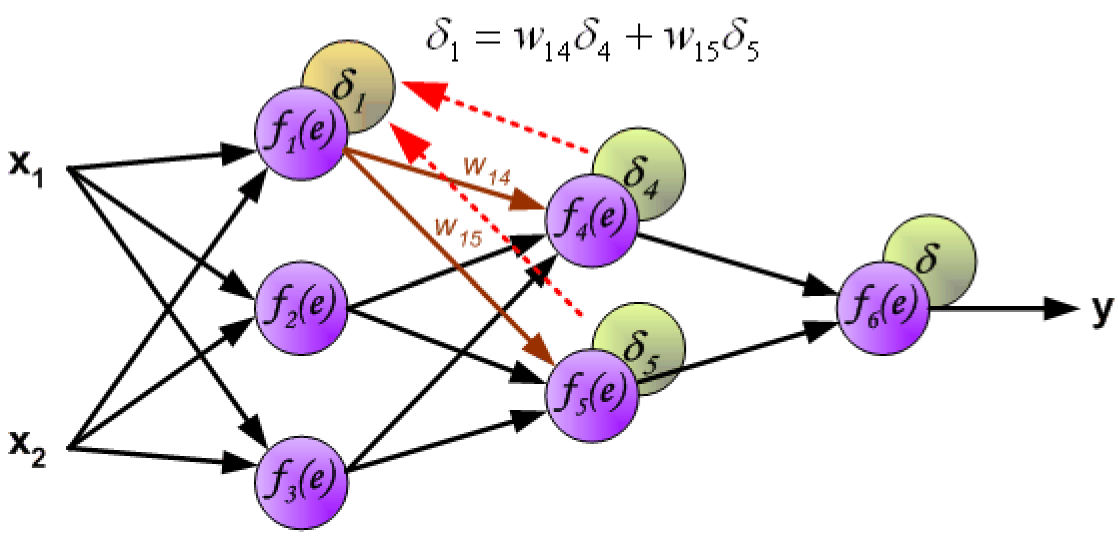

Figure 6 - Back propagating the error signal.

Once an error signal has been calculated for each neuron, the weights are updated accordingly. This involves taking the error signal, delta, times the derivative of the neuron activation function, times a learning rate which is manually set. This total value represents how much the weights should be updated, and is simply added to the old weight. When the activation function is a sigmoid function, the derivative is simple, and is usually pre-calculated to save on computational time.

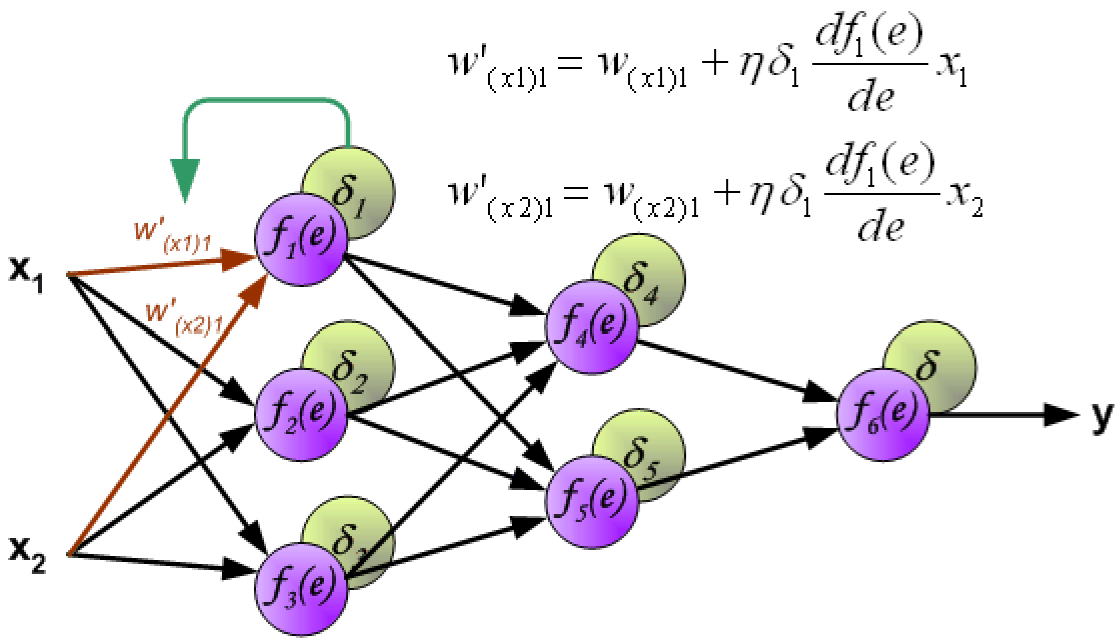

Figure 7 - Updating the weights using the backpropagated error signal.

In this fashion, the connections between every neuron are updated. An epoch is defined as feeding forward the entirety of a dataset, and performing backpropagation. Thus, a network is trained over multiple epochs, until it converges to some acceptable error rate. Some of the most widely used deep learning models are built upon the concept of neural networks. The structures of the networks are changed, but the end goal is the same, having the system learn an optimal set of weights for the task at hand. Deep learning networks are usually trained with backpropagation, with each layer from the input layer onwards being trained with backpropagation to optimize the weights.

ARCHITECTURES

Deep Learning is an encompassing term that contains a broad array of machine learning techniques and methods. The common thread between all of these techniques is that they all are hierarchical in nature, and they all use some non-linear information processing stage. Li Deng, a leading researcher in the field, suggests three broad categories of architectures (Deng). We fill focus on Generative Deep Architectures, which are the most commonly used for any high-dimensionality problem. (Such as speech analysis, text analysis, and so on).

Generative Deep Architectures

Generative architectures are deep learning techniques that model how the observed data were generated in order to classify a signal. They are “intended to characterize the high-order correlation properties of the observed or visible data for pattern analysis or synthesis purposes, and/or characterize the joint statistical distributions of the visible data and their associated classes (Deng).”

Deep learning models within this category are associated with unsupervised pre-training; where the network is trained without labelled data so it can learn features without any outside assistance. This is an especially useful application for natural language processing. One of the largest hurdles in classical machine learning is feature selection.

De-noising Autoencoders

Autoencoders are the most common type of deep models within this set. They are artificial neural networks that are used to learn “embeddings”. Structurally, they are the exact same as a feed-forward neural network, with the only difference being that the target values are set to be equal to the inputs.

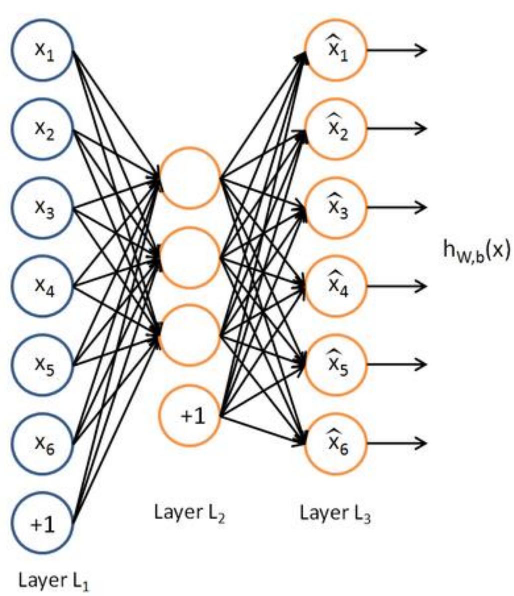

Figure 8 - Typical autoencoder.

The network learns to efficiently represent input features, finding the best weights that can recreate the inputs. These weights, the embeddings, are efficient coding of features; they are compressed representations of a set of data. This is accomplished by having a relatively small hidden layer, which forces the network to learn compressed representations (Hinton). Autoencoders are used for dimensionality reduction, and thus make high-dimensionality problems, like natural language processing tasks, more tractable to compute.

Such networks are trained in a greedy layer-by-layer fashion, where the network is trained in an unsupervised fashion (such as by feeding in the entirely of Wikipedia) so that it learns word representations, and then a soft-max layer or a logistic layer is stacked onto the pre-trained layers, and entirety of network is trained again using supervised data (Socher).

As an example, for natural language processing, the unsupervised training is accomplished by feeding in correct sentences. Sentences that have been "corrupted" with the insertion of a random word are also fed in. A scoring function is defined that scores how good the context of a sentence is, and a cost function is defined that tries to maximize the score of sentences with good context, whilst minimizing the score of sentences with bad context (Socher). A common scoring function used is the Kullback-Leibler distance between the original inputs and the reconstructed inputs. Once a layer has been trained using this method, uncorrupted embeddings are fed into the layer, and the outputs are used as the inputs to train the next layer.

Deep Belief Networks (Restricted Boltzmann Machines)

A Boltzmann is another type of neural network. It is a stochastic network where every neuron is symmetrically connected to every other neuron, both visible and hidden. The network is trained until it reaches equilibrium, which is defined as when the probability of the states of the network is given by the Boltzmann-Gibbs distribution (Chen).

However, they are very difficult to train, and are intractable to compute for any complex problem. A variant of Boltzmann machine, called the Restricted Boltzmann machine, can be efficiently trained, and is used in real-world applications. Restricted Boltzmann machines restrict the network such that hidden units can only connect to visible units, and no neurons within each layer can connect to each other.

Deep Belief Networks are formed by stacking Restricted Boltzmann machines, and are once more, used as a method to learn embeddings of high-dimensionality data. Greedy layer-by-layer pre-training is also employed to create a deep belief network. Each layer is trained until it reaches equilibrium as defined above. Once the layer is trained, the activities of the hidden units within that layer (the values of the hidden units) can be used to train the next Restricted Boltzmann machine layer (Ng). Thus, the deep network can learn efficient embeddings, and efficiently reduce the dimensionality of high-dimensional data. As before, these embeddings can be used to initialize a supervised training step, where a logistic or soft-max layer is used to feed in labelled data and train the network. A deep belief network is illustrated on the next page.

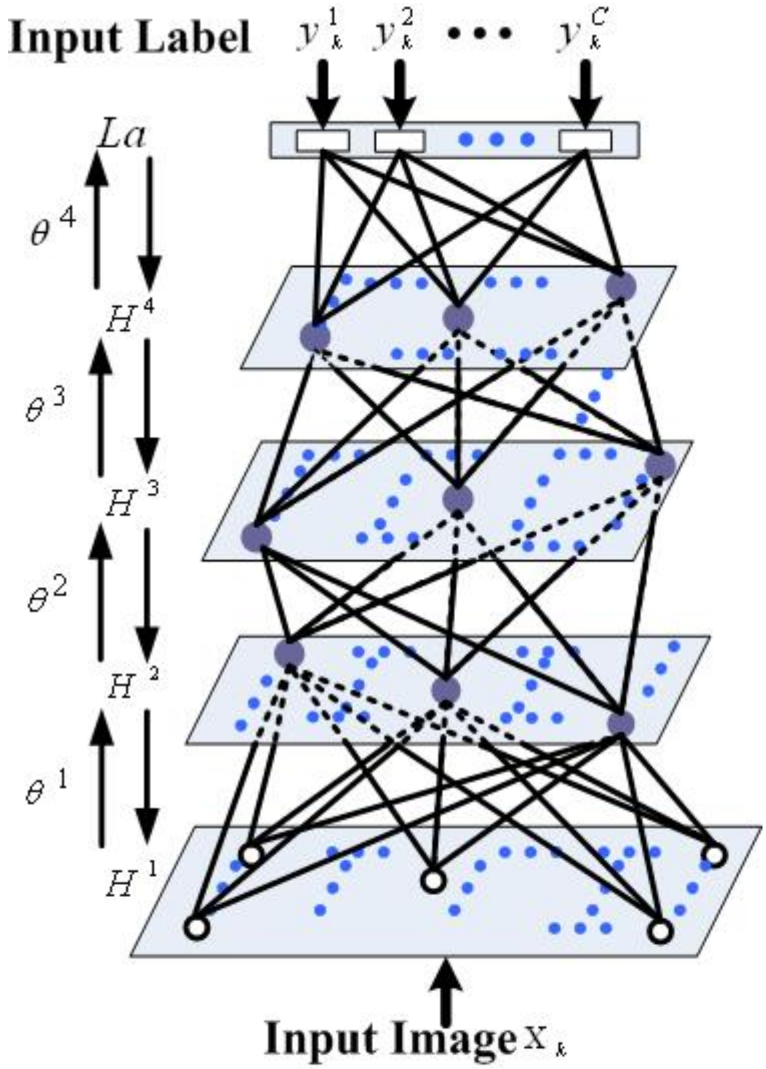

Figure 9 - Example implementation of a deep belif network for image recognition.

APPLICATIONS

Natural Language Processing

Deep learning has been applied quite successfully to a number of natural language processing tasks. Collobert and Weston, for example, created a deep belief network in 2008 that was able to simultaneously solve a multitude of classical NLP problems, such as chunking, named entity tagging, semantic identification, part-of-speech tagging, and similar word identification.

Another important and proven application of deep learning has been dimensionality reduction and feature selection for NLP tasks. Classically, features, words in this case, had to be manually selected in some fashion. Using an entire vocabulary was next to impossible in real-world applications, since a text with a million words would have resulted in sparse vectors with a length of one million. Thus, representing two simple sentences would, in this hypothetical, involve using two very sparse vectors, both of length one million. Therefore, features were selected in some fashion, and manually crafted to generate desirable results.

Deep models were shown to be suitable for both of the abovementioned tasks by Collobert and Weston. They used unsupervised learning, as described in the autoencoder section above, to take high dimensional spare vectors, and turn them into low-dimensional vectors that still proved to be very useful. These are the “word embeddings” that were described before. These embeddings were able to both capture syntactic and semantic similarities, and naturally clustered similar words and concepts together. For example, it was shown that feeding in the entirety of Wikipedia resulted in European cities clustering together, and another distinct cluster forming from American cities, and so on. Embeddings also allow for a natural way to select features, instead of manually selecting words, the network can learn to come to the most salient features via unsupervised training on a very large corpus.

Last, due in part to the abovementioned traits that make deep models suitable for NLP, deep models were shown to work very well when analyzing sentiment at a sentence-level. Richard Socher and his team created a deep learning model called a “Recursive Neural Tensor Network” that focused on using phrases, and their context in sentences, to classify sentiment (Socher).

They achieved results that were almost 10% better than using the “bag-of-words” model to represent features. (Which are the sparse vectors described before).

Language Modeling

One of the most interesting and successful applications of deep learning has been deep learning to language. Google, for example, has used deep learning models that can learn, without any supervision, how to accurately translate text between two different languages. The model works by analyzing how words are used in the two different languages, and representing those relationships as two-dimensional vectors (Derrick). These vectors were essentially embeddings, as described before.

The team at Google found that similar concepts between the two languages shared the same geometric arrangements (when the vectors were plotted). More so, they found that they could use these

similarities to accurately translate between languages. For example, the word horse in the English vector-space was similarly arranged as the word for horse in Spanish (“caballo”) in the Spanish vector-space.

Interestingly, they also found that they could find similar concepts between languages, even if the translation wasn’t entirely accurate. For example, for the Spanish translation of the word “empire”, the model suggested words that were conceptually similar, such as “dictatorship” and “imperialism”.

Computer Vision

Deep learning has also proven to be well-suited for computer vision and image recognition tasks. One of the key reasons is because of the various models’ ability to learn features without supervision. As detailed before, a deep learning system can hierarchically learn simple lines, then shapes, then complex objects, and so forth until it can recognize a complex set of items within an image. The concept employed is very similar to the se

Google once more is one of the key researchers within this area. The firm uses deep learning to improve its’ image search algorithm, and its image “reverse search”, which allows users to input an image and find similar images. The concept is very similar to the one used when using deep learning to analyze sentiment at a sentence level. An image is broken up into pieces, and the deep learning model learns to build up these pieces to correctly recognize an object, such as differentiating a building from a background.

CONCLUSION

Deep learning has proven itself to be suitable for almost every machine learning task. Deep learning models have become the best performing models in different fields, for vastly different tasks. Within computer vision, speech recognition, and natural language processing, deep learning models have outperformed previous cutting-edge methods by a large margin.

Their power stems from the hierarchical learning process they can employ, and the ability to take very high dimensional data and effectively represent them in lower dimensional space. Unsupervised pre-training has proven to be one of the most important parts of deep learning, and one of the key reasons why they outperform other machine learning models. Unsupervised pre-training allows deep learning models to learn features and efficient representations that drastically improve results. For example, in natural language processing, unsupervised pre-training has resulted in networks that can cluster words on both syntactic and semantic meaning, all from the data contained in context. This has drastically improved results with tasks such as sentiment analysis and similar word comparisons, all without requiring any new labelled training data.

Deep learning is not the panacea that will solve every difficult machine learning tasks out there, but it is certainly a step closer. They, more so than normal machine learning techniques, are able to continuously improve with the introduction of new data. Advances in computing will also improve them, as more and more complex networks become feasible, allowing for more advanced representations to be learned.

REFERENCES/CITATIONS

Chen, Edwin. “Introduction to Restricted Boltzmann Machines.”
http://blog.echen.me/2011/07/18/introduction-to-restricted-boltzmann-machines/

Collobert and Weston. “A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning.”
http://icml2008.cs.helsinki.fi/papers/391.pdf

Deng, Li. "A Tutorial Survey of Architectures, Algorithms, and Applications for Deep Learning."
http://research.microsoft.com/pubs/204048/APSIPA-Trans2013-revised-final.pdf

Google. "Speech Processing - Research at Google." Speech Processing - Research at Google
http://research.google.com/pubs/SpeechProcessing.html

Harris, Derrick. “How deep learning can teach computers Spanish without a tutor.”
http://gigaom.com/2013/09/26/how-deep-learning-can-teach-computers-spanish-without-a-tutor/

Hinton, Geoffrey. Salakhutdinov, R. R. “Reducing the Dimensionality of Data with Neural Networks”
http://www.cs.toronto.edu/~hinton/science.pdf

Ng, Andrew. Ngiam, Jiquan. Foo, Chuan. Mai, Yifan. Suen, Caroline. “Unsupervised Feature Learning and Deep Learning Tutorial.”
http://ufldl.stanford.edu/wiki/index.php/UFLDL_Tutorial

Socher, Richard. Perelygin, Alex. Wu, Jean. Chuang, Jason. Manning, Christopher. Ng, Andrew. Potts, Christopher. “Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank.”
http://nlp.stanford.edu/~socherr/EMNLP2013_RNTN.pdf

Socher, Richard. “Deep Learning for NLP without Magic.”
http://www.socher.org/index.php/DeepLearningTutorial/DeepLearningTutorial

IMAGES FROM:

http://home.agh.edu.pl/~vlsi/AI/backp_t_en/backprop.html

http://theanalyticsstore.com/deep-learning/

http://ufldl.stanford.edu/wiki/index.php/UFLDL_Tutorial